IN THE CLAIMS:

1. (Currently amended) A method of verifying symbolic data for ~~loaded~~ modules that are loaded in a computer system, comprising:

reading trace data for a module, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program;

comparing the trace data with ~~module~~ symbolic data stored in a merged symbol file, wherein the symbolic data is data representative of human readable alphanumeric text associated with modules that are loaded in the computer system; and

verifying the symbolic data in response to a determination that the trace data matches the ~~module~~ symbolic data in the merged symbol file based on one or more predetermined criteria.

2. (Original) The method of claim 1, wherein the one or more predetermined criteria include one or more of a checksum, a timestamp, a fully qualified path, and a segment size.

3. (Original) The method of claim 1, wherein the trace data is read from a trace buffer.

4. (Original) The method of claim 1, wherein the trace data is read from a trace file written to a storage device.

5. (Original) The method of claim 1, wherein the reading, comparing and verifying steps are performed dynamically as trace data is written to a trace buffer.

6. (Original) The method of claim 1, wherein the reading, comparing and verifying steps are performed at a remote time from when the trace data is written to a trace file.

7. (Currently amended) The method of claim 1, wherein comparing the trace data with ~~module~~ symbolic data in a merged symbol file includes comparing a checksum and timestamp in the trace data with a checksum and timestamp in the ~~module~~ symbolic data in the merged symbol file.

8. (Currently amended) The method of claim 7, wherein comparing the trace data with ~~module~~ symbolic data in a merged symbol file further includes comparing a fully qualified path in the trace data with a fully qualified path in the ~~module~~ symbolic data, if the checksum and timestamp in the trace data does not match the checksum and timestamp in the ~~module~~ symbolic data or the checksum and timestamp in the trace data is not available.

9. (Currently amended) The method of claim 8, wherein comparing the trace data with ~~module~~ symbolic data in a merged symbol file further includes comparing a segment length in the trace data with a segment length in the ~~module~~ symbolic data, if the fully qualified path in the trace data does not match the fully qualified path in the ~~module~~ symbolic data.

10. (Original) The method of claim 1, wherein the one or more criteria have an associated priority.

11. (Original) The method of claim 10, wherein the one or more criteria include checksum and timestamp, a fully qualified path, and a segment size, and wherein the checksum and timestamp has a highest priority and the segment size has a lowest priority.

12. (Currently amended) The method of claim 1, wherein the merged symbol file includes a plurality of module entries and wherein comparing the trace data with ~~module~~ symbolic data in a merged symbol file includes identifying a module entry that is a best match with the trace data.

13. (Original) The method of claim 12, wherein identifying a module entry that is a best match with the trace data includes comparing the trace data to each of the plurality of module entries and identifying one of the plurality of module entries as a best match based on which of the one or more criteria is used to verify the module entry.

14. (Original) The method of claim 1, wherein the trace data includes redundant information identifying a module for each segment of the module.

15. (Original) The method of claim 1, wherein the redundant information includes at least one of module checksum, module timestamp and module fully qualified path.

16. (Currently amended) A method of displaying data for analyzing a performance trace of a computer application, comprising:

reading module trace data from a trace file, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program;

reading module symbolic data from a symbolic data file;

verifying that the module symbolic data corresponds to the module trace data;

correlating the module symbolic data with the module trace data to generate correlated data; and

displaying the correlated data.

17. (Original) The method of claim 16, wherein the step of verifying includes:

comparing the trace data with the module symbolic data; and

verifying that the trace data matches the module symbolic data based on one or more predetermined criteria.

18. (Original) The method of claim 17, wherein the one or more predetermined criteria include one or more of a checksum, a timestamp, a fully qualified path, and a segment size.

19. (Original) The method of claim 16, wherein the trace data is read from a trace buffer.

20. (Original) The method of claim 16, wherein the trace data is read from a trace file written to a storage device.

21. (Original) The method of claim 17, wherein the one or more criteria have an associated priority.

22. (Original) The method of claim 21, wherein the one or more criteria include checksum and timestamp, a fully qualified path, and a segment size, and wherein the checksum and timestamp has a highest priority and the segment size has a lowest priority.

23. (Original) The method of claim 16, wherein the merged symbol file includes a plurality of module entries and wherein comparing the trace data with module symbolic data in a merged symbol file includes identifying a module entry that is a best match with the trace data.

24. (Original) The method of claim 23, wherein identifying a module entry that is a best match with the trace data includes comparing the trace data to each of the plurality of module entries and identifying one of the plurality of module entries as a best match based on which of the one or more criteria is used to verify the module entry.

25. (Original) The method of claim 16, wherein the trace data includes redundant information identifying a module for each segment of the module.

26. (Original) The method of claim 16, wherein the redundant information includes at least one of module checksum, module timestamp and module fully qualified path.

27. (Currently amended) An apparatus for verifying symbolic data for ~~loaded~~ modules that are loaded in a computer system, comprising:

a trace data storage device;

a merged symbol file storage device; and

a processor coupled to the trace data storage device and the merged symbolic data storage device, wherein the processor reads trace data for a module from the trace data storage device, compares the trace data with ~~module~~ symbolic data stored in a merged symbol file read from the merged symbol file storage device, wherein the symbolic data is data representative of human readable alphanumeric text associated with modules that are loaded in the computer system, and verifies the symbolic data in response to a determination that the trace data matches the ~~module~~ symbolic data in the merged symbol file based on one or more predetermined criteria, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program.

28. (Original) The apparatus of claim 27, wherein the one or more predetermined criteria include one or more of a checksum, a timestamp, a fully qualified path, and a segment size.

29. (Original) The apparatus of claim 27, wherein the trace data storage device is a trace buffer.

30. (Currently amended) The apparatus of claim 27, wherein the processor reads the trace data, compares the trace data with ~~module~~ symbolic data and verifies that the trace data matches the ~~module~~ symbolic data dynamically as trace data is written to the trace data storage device.

31. (Currently amended) The apparatus of claim 27, wherein the processor reads the trace data, compares the trace data with ~~module~~ symbolic data and verifies that the trace data matches the ~~module~~ symbolic data at a remote time from when the trace data is written to the trace data storage device.

32. (Currently amended) The apparatus of claim 27, wherein the processor compares the trace data with ~~module~~ symbolic data in a merged symbol file by comparing a checksum and timestamp in the trace data with a checksum and timestamp in the ~~module~~ symbolic data in the merged symbol file.

33. (Currently amended) The apparatus of claim 32, wherein the processor compares the trace data with ~~module~~ symbolic data in a merged symbol file by further comparing a fully qualified path in the trace data with a fully qualified path in the ~~module~~ symbolic data, if the checksum and timestamp in the trace data does not match the checksum and timestamp in the ~~module~~ symbolic data or the checksum and timestamp in the trace data is not available.

34. (Currently amended) The apparatus of claim 33, wherein the processor compares the trace data with ~~module~~ symbolic data in a merged symbol file by further comparing a segment length in the trace data with a segment length in the ~~module~~ symbolic data, if the fully qualified path in the trace data does not match the fully qualified path in the ~~module~~ symbolic data.

35. (Original) The apparatus of claim 27, wherein the one or more criteria have an associated priority.

36. (Original) The apparatus of claim 35, wherein the one or more criteria include checksum and timestamp, a fully qualified path, and a segment size, and wherein the checksum and timestamp has a highest priority and the segment size has a lowest priority.

37. (Currently amended) The apparatus of claim 27, wherein the merged symbol file includes a plurality of module entries and wherein the processor compares the trace data with ~~module~~ symbolic data in a merged symbol file by identifying a module entry that is a best match with the trace data.

38. (Original) The apparatus of claim 37, wherein the processor identifies a module entry that is a best match with the trace data by comparing the trace data to each of the plurality of module entries and identifying one of the plurality of module entries as a best match based on which of the one or more criteria is used to verify the module entry.

39. (Original) The apparatus of claim 27, wherein the trace data includes redundant information identifying a module for each segment of the module.

40. (Original) The apparatus of claim 27, wherein the redundant information includes at least one of module checksum, module timestamp and module fully qualified path.

41. (Currently amended) An apparatus for displaying data for analyzing a performance trace of a computer application, comprising:

a trace data storage device;

a symbolic data storage device; and

a processor coupled to the trace data storage device and the symbolic data storage device, wherein the processor reads module trace data from the trace data storage device, reads module symbolic data from the symbolic data storage device, verifies that the module symbolic data corresponds to the module trace data, correlates the module symbolic data with the module trace data to generate correlated data, and displays the correlated data on a display device, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program.

42. (Currently amended) A computer program product in a computer readable medium for verifying symbolic data for ~~loaded~~ modules, comprising:

first instructions for reading trace data for a module, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program;

second instructions for comparing the trace data with ~~module~~ symbolic data in a merged symbol file, wherein the symbolic data is data representative of human readable alphanumeric text associated with modules that are loaded in the computer system; and

third instructions for verifying that the trace data matches the ~~module~~ symbolic data in the merged symbol file based on one or more predetermined criteria.

43. (Original) The computer program product of claim 42, wherein the one or more predetermined criteria include one or more of a checksum, a timestamp, a fully qualified path, and a segment size.

44. (Currently amended) The computer program product of claim 42, wherein the second instructions for comparing the trace data with ~~module~~ symbolic data in a merged symbol file include instructions for comparing a checksum and timestamp in the trace data with a checksum and timestamp in the ~~module~~ symbolic data in the merged symbol file.

45. (Currently amended) The computer program product of claim 44, wherein the second instructions for comparing the trace data with ~~module~~ symbolic data in a merged symbol file further include instructions for comparing a fully qualified path in the trace data with a fully qualified path in the ~~module~~ symbolic data, if the checksum and timestamp in the trace data does not match the checksum and timestamp in the ~~module~~ symbolic data or the checksum and timestamp in the trace data is not available.

46. (Currently amended) The computer program product of claim 45, wherein the second instructions for comparing the trace data with ~~module~~ symbolic data in a merged symbol file further include instructions for comparing a segment length in the trace data with a segment length in the ~~module~~ symbolic data, if the fully qualified path in the trace data does not match the fully qualified path in the ~~module~~ symbolic data.

47. (Currently amended) The computer program product of claim 42, wherein the merged symbol file includes a plurality of module entries and wherein the second

instructions for comparing the trace data with ~~module~~ symbolic data in a merged symbol file include instructions for identifying a module entry that is a best match with the trace data.

48. (Original) The computer program product of claim 47, wherein the instructions for identifying a module entry that is a best match with the trace data include instructions for comparing the trace data to each of the plurality of module entries and identifying one of the plurality of module entries as a best match based on which of the one or more criteria is used to verify the module entry.

49. (Currently amended) A computer program product in a computer readable medium for displaying data for analyzing a performance trace of a computer application, comprising:

first instructions for reading module trace data from a trace file, wherein the trace data is obtained from a trace of an execution of a computer program of which the module is a part, and wherein the trace data is provided in terms of machine code corresponding to the computer program;

second instructions for reading module symbolic data from a symbolic data file;

third instructions for verifying that the module symbolic data corresponds to the module trace data;

fourth instructions for correlating the module symbolic data with the module trace data to generate correlated data; and

fifth instructions for displaying the correlated data.